VI Simpósio de Iniciação Científica 2023
Ciência como ferramenta de transformação da sociedade

Fortalecimento da Formulação do Problema do Empacotamento Bidimensional em Contêineres na Presença de Conflitos

Ana Clara Nascimento dos Santos[1] (IC), Prof. Dr. Pedro H. D. B. Hokama (PQ)[1], Prof. Dr. Mário César San Felice (PQ)[2]
[1] Universidade Federal de Itajubá, [2]Universidade Federal de São Carlos

Palavras-chave: Otimização Combinatória, Branch and Cut, Programação Linear Inteira.

Introdução

Em empresas de diversos setores, a gestão de processos logísticos desempenha um papel crucial. Ela visa a organização eficaz de produtos, melhorando a utilização do espaço em seu transporte e armazenamento e, consequentemente, reduzindo custos operacionais. Entretanto, podem surgir desafios adicionais quando esses processos envolvem produtos que não podem ser alocados juntos devido ao risco de contaminação ou à incompatibilidade entre materiais, tais como substâncias explosivas, inflamáveis, corrosivas ou tóxicas, como destaca por Capua et al. (2015)

O Problema do Empacotamento Bidimensional em Contêineres na Presença de Conflitos (*Two-dimensional Bin Packing Problem With Conflicts - 2BPPC*) consiste em alocar um conjunto de itens retangulares de tamanhos variados no menor número possível de recipientes idênticos, respeitando as restrições de conflito entre os itens. Esse problema é classificado como NP-difícil, ou seja, ele não pode ser solucionado na otimalidade em tempo polinomial a menos que P=NP. O 2BPPC pode ser decomposto em dois problemas. O primeiro é a sua versão unidimensional (*Bin Packing Problem With Conflicts - BPPC*) que tem o objetivo de agrupar os itens no menor número de contêineres possível respeitando os conflitos. O segundo é o Problema do Empacotamento Bidimensional Ortogonal (*Two-dimension Orthogonal Packing Problem - 2D-OPP*) que decide se há um empacotamento ou não em apenas um contêiner para um dado conjunto de itens. O objetivo deste trabalho é explorar a integração de abordagens exatas baseadas em Programação Linear Inteira (PLI) e Programação por Restrição (CP) para resolver o 2BPPC.

Metodologia

Entre as principais referências desta pesquisa estão Gendreau et al. (2004) que abordaram o BPP unidimensional com conflitos por meio de heurísticas, cliques e limitantes inferiores, Capua et al. (2015) que trabalharam com essa mesma variação do problema e empregaram uma abordagem exata baseada na formulação do problema de cobertura por conjuntos e Carlier et al. (2007) apresentaram novos procedimentos de redução e limitantes inferiores para a variação do 2BPP sem conflitos. O foco dessa pesquisa está na resolução do problema mestre BPPC, utilizando PLI e várias restrições para fortalecer o modelo. Para verificar a viabilidade de um empacotamento, resolvendo um 2D-OPP, foi utilizado um resolvedor de Programação por Restrições desenvolvido por Boulos et al. (2023).

Descrição e Formulação do 2BPPC

O 2BPPC pode ser descrito da seguinte forma: Dado V um conjunto de n itens, com $i \in V$ possuindo altura h_i e largura w_i e o conjunto Q de bins idênticos de altura H e largura W. Um grafo não direcionado G = (V, E), no qual os vértices são os itens, e as arestas o conflito entre eles. Em uma solução para esse problema, cada item i deve ser alocado em exatamente um bin $k \in Q$, não sendo permitido que dois itens conflitantes sejam empacotados no mesmo recipiente. Além disso, a orientação de cada item é fixa, não sendo permitidas rotações e eles não devem se sobrepor ou exceder a borda do recipiente. O objetivo é encontrar uma solução que minimize o número de bins utilizados.

Implementamos uma adaptação da formulação proposta por Gendreau et al. (2004) para o BPPC, nela temos as variáveis binárias β_k, que indica se o bin k está sendo utilizado, considerando o valor 1, ou 0 caso contrário, e $\alpha_{i,k}$ que indica se o item i está presente no bin k, assumindo valor 1 se está e 0 caso contrário. Considere $\mathcal{S}$ como a coleção de todos os subconjuntos de V que cabem simultaneamente em um bin.

$$\text{Minimize: } \sum_{k=1}^{n} \beta_k \tag{1}$$

$$\text{Sujeito à: } \sum_{i=1}^{n} w_i h_i \alpha_{i,k} \leq WH\beta_k, \qquad k \in \{1,\ldots,n\} \tag{2}$$

$$\sum_{k=1}^{n} \alpha_{i,k} = 1, \qquad i \in \{1,\ldots,n\} \tag{3}$$

$$\alpha_{i,k} + \alpha_{j,k} \leq 1, \qquad (i,j) \in E, k \in \{1,...,n\} \quad (4)$$

$$\sum_{i \in S} \alpha_{i,k} \leq |S| - 1, \qquad S \notin \mathcal{S}, k \in \{1,...,n\} \quad (5)$$

$$\beta_k \in \{0,1\}, \qquad k \in \{1,...,n\} \quad (6)$$

$$\alpha_{i,k} \in \{0,1\}, \qquad i \in \{1,...,n\}, k \in \{1,...,n\} \quad (7)$$

O objetivo definido por (1) consiste em minimizar o número de bins utilizados. A restrição (2) garante que a soma das áreas dos itens seja menor que a área do bin e que os itens só serão empacotados em bins que forem utilizados na solução, (3) certifica que cada item será alocado exatamente em um recipiente, e (4) assegura que dois itens que possuem conflito (aresta entre eles) não podem ser alocados no mesmo bin. A restrição (5) garante que os itens sejam empacotáveis, isto é, que os conjuntos de itens que excedem os limites do recipiente quando empacotados juntos não serão alocados no mesmo bin. As restrições (6) e (7) apontam o domínio das variáveis binárias.

É importante ressaltar que o número de conjuntos não empacotáveis é exponencial, o que impossibilita que todas as restrições (5) sejam inseridas, recorremos então ao uso de Lazy Constraints para relaxar essa restrição e adicioná-la sob demanda. Nesse ponto, surge o 2D-OPP, que verifica se o empacotamento de um conjunto de itens é factível ou não.

Para fortalecer a formulação (1) - (7) apresentamos restrições baseadas em limitantes, pré-processamentos e cortes, as principais delas serão descritas a seguir:

Fixando item (F): o primeiro item da instância é pré-alocado no recipiente 1, o que elimina opções simétricas de empacotamento do item $i = 1$ no modelo, logo não são testadas as possibilidades desse item ser alocados aos outros bins disponíveis.

$$\alpha_{1,1} = 1$$

Bins Consecutivos (B): Condiciona o uso do bin *k*, ao bin anterior *k-1* já ter sido utilizado. Essa restrição elimina simetrias, já que uma solução que utilizasse por exemplo os bins {1, 3, 7} tem uma solução equivalente que utiliza apenas os bins {1, 2, 3} já que os bins são idênticos.

$$\beta_{k-1} \geq \beta_k \qquad k \in \{2, \ldots, n\}$$

Restringindo Bins dos Itens (I): Restringe os bin aos quais os itens podem ser alocados. Serão considerados apenas metade dos itens, para evitar que restrições fracas sejam adicionadas. Essa restrição obriga o item $i \in \{2, \ldots, n/2\}$ a estar nos um dos i primeiros bins.

$$\sum_{k=1}^{i} \alpha_{i,k} = 1 \qquad i \in \{2, \ldots, n/2\}$$

Restringindo Itens Largos ou Altos (R2): Decompõe os itens em dois conjuntos; altos e largos, em função da metade da altura e da largura do Bin, tal que $A_{altos} = \{i \in V: h_i > H/2\}$ e $A_{largos} = \{i \in V: w_i > W/2\}$. Em seguida, percorre todos os bins, de forma que a altura dos itens classificados como largos é acumulada e deve corresponder a no máximo a altura do bin (8), pois os itens largos ocupam mais da metade do bin, não permitindo dois deles lado a lado, logo apenas uma pilha pode ser formada e deve ser limitada a altura do recipiente. Ocorre o mesmo para os itens altos, como são limitados a apenas uma fileira horizontal, suas larguras são somadas e restritas a no máximo a largura do recipiente (9). Essa restrição poda as tentativas de combinações que excederiam as dimensões do bin.

$$\sum_{k=1}^{n} \alpha_{i,k} h_i \leq H \qquad i \in A_{largos} \quad (8)$$

$$\sum_{k=1}^{n} \alpha_{i,k} w_i \leq W \qquad i \in A_{altos} \quad (9)$$

Outras três restrições seguem a mesma lógica, **Restringindo Itens (R3), Restringindo Itens (R4)** e **Restringindo Itens (Rk)** entretanto R3 e R4 consideram a discretização das dimensões do recipiente em 3 e 4 faixas, respectivamente e Rk testa todas as possíveis divisões com $k \in \{2,..., n\}$ e escolhe a divisão que resulte na maior altura acumulada possível em cada uma das k-1 pilhas disponíveis (10). De forma análoga, um k que maximize a largura acumulada e esteja entre 2 e H é encontrado.

$$\max_k \left\{ \frac{\sum_{i=1} peso(i,k) h_i}{(k-1)} \right\} \quad (10)$$

$$peso(i,k) = \left\lfloor \frac{w_i}{W/k + \epsilon} \right\rfloor \quad i \in \{1,...,n\}\ k \in \{2,...,W\} \quad (11)$$

$$\sum_{i=1} peso(i,k) h_i \leq (k-1)H \quad i \in A_{largos} \quad (12)$$

Adicionando Conflitos Por Incompatibilidade De Dimensões (D): Percorre os itens, comparando cada um deles com todos os itens que o sucedem, se a soma das larguras e alturas for maior que a dimensão do bin, não podem ser empacotados juntos. Portanto, um conflito é adicionado ao modelo, o que evita que empacotamentos infactíveis devido às restrições de tamanho do bin sejam testadas, podando o espaço de busca.

Adicionando Conflitos por Cliques (C): Nessa abordagem, foi utilizada a biblioteca Cliquer (Niskanen, 2002) adaptada para C + +. Uma clique em um grafo não-direcionado corresponde a um subconjunto de seus vértices, tal que cada dois vértices são adjacentes. No

problema abordado, uma clique representa itens que não podem ser empacotados no mesmo bin, já que possuem conflitos com todos os outros que também compõe a clique. As cliques maximais são encontradas através de funções dessa biblioteca e por fim, é adicionada a restrição de que itens da mesma clique não podem ser empacotados no mesmo recipiente, fortalecendo o modelo.

Adicionando Lower Bound (L1): Utilizamos o limitante inferior para o número de bins analisado por Carlier et al. (2007), no qual o conjunto dos itens V é decomposto em conjuntos de grandes, médios, altos, largos e pequenos em função dos inteiros p e q, tal que $1 \leq p \leq H/2$ *e* $1 \leq q \leq W/2$. Após a classificação dos itens, é feito o cálculo de quantos quadrados do bin discretizado ele ocupa (m). Para cada combinação de p e q, o m de todos os itens, exceto os grandes, são somados e usados para obter o número de bins necessários para os itens menores. Por fim, esse valor é somado ao número de itens grandes e médios. Ao final, o maior Lower Bound é adicionado ao modelo.

Adicionando Upper Bound (L2): para definir um limite superior para nosso modelo, utilizamos uma abordagem na qual todos os itens primeiramente eram alocados em prateleiras, as quais possuem a largura idêntica a do bin, a altura do primeiro item ali colocado e herdam o conflito de cada um dos que ali estejam. Sempre que possível o item é colocado em uma prateleira já existente, quando ocorrem conflitos ou as dimensões disponíveis na prateleira são excedidas, uma nova é alocada. Após todos os itens serem destinados a alguma prateleira, estas são empacotadas nos bins, seguindo as mesmas restrições de tamanho e compatibilidade. Ao final, tem-se o número máximo de bins necessários para resolver o 2BPPC para aquela instância. Logo, não serão consideradas soluções que utilizem mais recipientes do que o número obtido como limitante.

Ordenando Decrescente (O): O número de conflitos de cada um dos itens é contabilizado e baseado nessas quantidades eles são ordenados de forma decrescente. Desse modo, os itens com mais conflitos serão alocados primeiro, o que evita desperdício de espaço em bins, que poderia ocorrer caso ele fosse alocado depois de diversos itens e não pudesse ser empacotado nos recipientes até então utilizados, devido a presença de um ou mais itens incompatíveis com ele.

Como mencionado anteriormente, o 2D-OPP é usado para verificar a viabilidade de um empacotamento de um conjunto de itens e foi solucionado utilizando o resolvedor de Programação por Restrições (CP) desenvolvido por Boulos et al.(2023).

Todos os modelos e algoritmos descritos foram implementados em C + + e compilados com G + + 11.3 em ambiente linux em um processador de 3.60GHz e 16GB de memória ram. O resolvedor de PLI foi o IBM Cplex 22.1 e o resolvedor de CP foi o IBM CP Optimizer 22.1, ambos do mesmo pacote.

Após a conclusão das melhorias propostas, procedeu-se com a integração das abordagens de Programação Linear Inteira (PLI) e Programação por Restrição (CP). Este processo envolveu alguns pequenos ajustes e adaptações no ambiente de desenvolvimento.

Logo após concluída a integração, iniciamos os testes que foram executados em 64 instâncias sendo: 48 apresentadas por de Queiroz e Miyazawa (2012) para o problema da mochila com conflitos, no caso do 2BPPC todos os itens de cada instância precisam ser empacotados e os valores dos itens foram ignorados. Outras 16 instâncias foram criadas modificando as originais, com o propósito de comparar como performavam os algoritmos quanto mais itens cabiam por contêiner.

Testar todas as combinações possíveis de melhorias desenvolvidas seria inviável, portanto optamos por agrupar algumas categorias de restrições, com o objetivo de verificar o tempo em que cada configuração consegue resolver as instâncias.

Resultados e discussão

Os resultados serão apresentados através de gráficos de *performance profiles* (Dolan e Moré, 2002), em que cada curva representa uma configuração do nosso algoritmo, o eixo vertical representa a porcentagem das instâncias resolvidas, e o eixo horizontal representa a razão em relação ao melhor tempo.

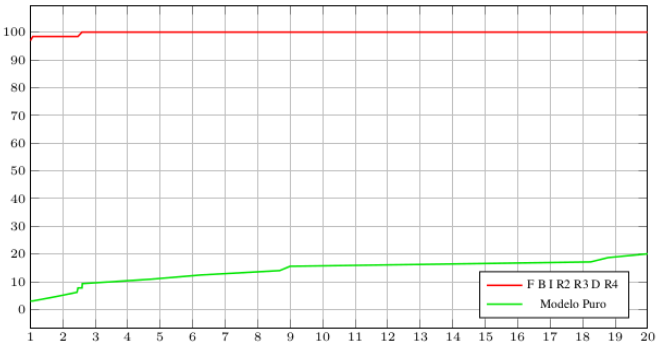

Figura 1 – Comparação dos resultados entre o modelo (1)-(7) sem nenhuma das melhorias propostas e adicionado o conjunto de melhorias F, B, I, R2, R3, D e R4.

A Figura 1 apresenta a comparação entre o modelo

básico, a formulação em PLI pura sem melhorias (linha verde) e a configuração com as melhorias F B I R2 R3 D R4 (linha vermelha), podemos notar que o impacto dessas melhorias foi significativo. A curva do modelo com melhorias começa perto de 100, o que significa que praticamente todas as instâncias foram resolvidas mais rapidamente por essa versão, enquanto na versão básica mesmo com 20 vezes o melhor tempo só era possível solucionar 20% das instâncias.

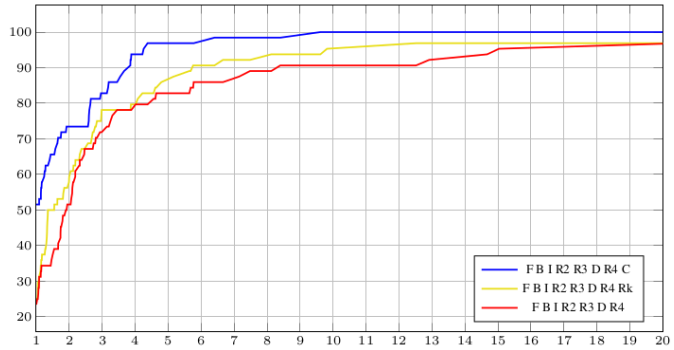

Figura 2 - Comparação dos resultados entre a configuração F B I R2 R3 D R4 e dessa acrescida da melhoria Rk, depois com a melhoria C. Em todos os casos o problema do empacotamento foi resolvido com um modelo de CP.

A Figura 2 mostra a comparação entre o F B I R2 R3 D R4 da figura anterior acrescidas das melhorias C e Rk separadamente. Observamos que com a ativação de Rk melhoramos o desempenho, e com a ativação das cliques conseguimos melhorias ainda mais significativas, sendo que foi a mais rápida desse conjunto em 50% das instâncias.

Conclusões

Os resultados obtidos mostram que para o conjunto de instâncias testadas as técnicas aplicadas foram muito significativas. Diversas instâncias que não eram resolvidas em 3600 segundos passaram a ser resolvidas em 0.03 segundos ou menos. Somando os tempos para resolver as 64 instâncias, o modelo básico levou quase 44 horas (o tempo máximo de cada instância é 3600 segundos) enquanto na melhor configuração esse tempo foi de 30 minutos, sendo que praticamente todo esse tempo foi gasto nas instâncias 53 e 55.
Concluímos então que as melhorias adaptadas e propostas foram muito efetivas. Nos próximos passos dessa pesquisa pretendemos aplicar mais algumas melhorias, como por exemplo o procedimento de lifting dos itens. Outras integrações também podem ser interessantes, como a integração com o problemas de roteamento ou de dimensionamento de lotes.

Agradecimentos

Agradeço a FAPEMIG (Fundação de Amparo à Pesquisa do Estado de Minas Gerais) pela bolsa de Iniciação Científica através do programa PIBIC 2097/2022, aos professores Pedro Henrique Del Bianco Hokama e Mário César San Felice e à Universidade Federal de Itajubá.

Referências

BOULOS, Charbel Daher; HOKAMA, Pedro Henrique Del Bianco; SAN FELICE, Mário César. Padrões para o Problema do Empacotamento Bidimensional. In: **VI Simpósio de Iniciação Científica da Unifei**, 2023, Unifei.

CAPUA, Renatha et al. Um algoritmo heurístico para o problema de bin packing com conflitos. **Anais do XLVII Simpósio** Brasileiro **de Pesquisa Operacional**, 2015.

CARLIER, Jacques; CLAUTIAUX, François; MOUKRIM, Aziz. New reduction procedures and lower bounds for the two-dimensional bin packing problem with fixed orientation. **Computers & Operations Research**, v. 34, n. 8, p. 2223-2250, 2007.

DE QUEIROZ, Thiago Alves; MIYAZAWA, Flávio Keidi. Problema da mochila 0-1 bidimensional com restricoes de disjuncao. In: **Anais do XVI CLAIO/XLIV SBPO–Simpósio Brasileiro de Pesquisa Operacional/Congreso Latino-Iberoamericano de Investigación Operativa.** 2012. p. 1-12.

DOLAN, Elizabeth D.; MORÉ, Jorge J. Benchmarking optimization software with performance profiles. **Mathematical programming**, v. 91, p. 201-213, 2002.

GENDREAU, Michel; LAPORTE, Gilbert; SEMET, Frédéric. Heuristics and lower bounds for the bin packing problem with conflicts. **Computers & Operations Research**, v. 31, n. 3, p. 347-358, 2004**.**

NISKANEN, Sampo. Cliquer - **Routines For Clique Searching**. 2002. Disponível em https://users.aalto.fi/\~pat/cliquer.html. Acesso em 09/09/2022.